Algèbre linéaire de base

J. Erhel

Janvier 2014

Ce chapitre introduit les notations et les opérations de base sur l'algèbre des matrices. Il se termine par des notions de complexité et de performances et par la description des bibliothèques BLAS et LAPACK.

1 Espaces vectoriels et vecteurs

On note $\mathbb{R}^n$ un espace vectoriel de dimension n.
La base canonique de $\mathbb{R}^n$ est $(e_1, \ldots, e_n)$.
Un vecteur $x \in \mathbb{R}^n$, de composantes $x_1, \ldots, x_n$, est noté $x = (x_i)$.
Les vecteurs sont notés verticalement.

2 Matrices et vecteurs

Une matrice $A \in \mathbb{R}^{m \times n}$ est notée $A = (a_{ij})$.
Le j^{eme} vecteur colonne de A est $a_j = Ae_j$.
Un système de n vecteurs $u_1, \ldots, u_n \in \mathbb{R}^m$ est noté sous forme matricielle $U = (u_1 \ldots u_n) \in \mathbb{R}^{m \times n}$, avec u_j le j^{eme} vecteur colonne de U.
On note $Vect(U)$ le sous-espace vectoriel (sev) engendré par les vecteurs colonnes de U.
Si U est un système libre de k vecteurs, $Vect(U)$ est un sev de dimension k.

2.1 Matrices carrées et matrices particulières

Une matrice $A \in \mathbb{R}^{m \times n}$ est carrée d'ordre n si $n = m$.

La trace d'une matrice carrée A d'ordre n est la somme de ses éléments diagonaux : $tr(A) = \sum_{i=1}^{n} a_{ii}$.

Le déterminant d'une matrice carrée A d'ordre n est noté $det(A)$.

La matrice identité d'ordre k, dans $\mathbb{R}^{k \times k}$, vaut $I_k = (e_1 \ldots e_k)$.

Une matrice carrée D est diagonale si les seuls éléments non nuls sont sur la diagonale ; elle est notée $D = diag(d_i)$, où $d = (d_i)$ est le vecteur formé par les éléments diagonaux.

Une matrice carrée L triangulaire inférieure si les seuls éléments non nuls sont dans le triangle inférieur ; on définit de même une matrice carrée U triangulaire supérieure.

Une matrice tridiagonale a trois diagonales non nulles, une matrice bidiagonale a deux diagonales non nulles.

2.2 Opérations sur les matrices

L'ensemble des matrices $\mathbb{R}^{m\times n}$ est un espace vectoriel de dimension mn.

Soit $A \in \mathbb{R}^{m\times n}$ et $B \in \mathbb{R}^{n\times p}$; le produit $C = AB \in \mathbb{R}^{m\times p}$ est défini par $c_{ij} = \sum_{k=1}^{n} a_{ik}b_{kj}$.

L'ensemble des matrices carrées $\mathbb{R}^{n\times n}$ est un anneau.

L'anneau n'est pas commutatif (il existe A et B tels que $AB \neq BA$).

L'anneau n'est pas intègre (il existe des diviseurs de zéro : il existe A et B tels que $AB = 0$).

Une matrice carrée A est inversible s'il existe une matrice B telle que $AB = I_n$. Si B existe, alors $BA = I_n$, B est unique, c'est l'inverse de A et on note $B = A^{-1}$.

L'anneau n'est pas un corps (il existe des matrices non inversibles, dites singulières).

Le produit de deux matrices diagonales est une matrice diagonale.

Le produit de deux matrices triangulaires est une matrice triangulaire.

2.3 Matrices symétriques

La transposée A^T d'une matrice carrée A est la matrice obtenue en interchangeant les lignes et les colonnes. Soit $A = (a_{ij})$ et $B = A^T = (b_{ij})$, on a donc $b_{ij} = a_{ji}$.

Une matrice carrée A est symétrique si $A = A^T$.

Les matrices A^TA et AA^T sont symétriques.

On a $(A^T)^T = A$ et $(AB)^T = B^TA^T$.

2.4 Partitions par blocs

Une matrice par blocs est définie par une partition où les éléments scalaires sont regroupés dans des sous-matrices ; on note $A = A_{ij}$.

On définit les mêmes règles d'opérations, en respectant les dimensions dans les produits. Attention à l'ordre des blocs dans les produits.

3 Produit scalaire et normes vectorielles

Le produit scalaire de deux vecteurs x et y est $x^Ty = \sum_{i=1}^{n} x_iy_i$.

Soit $x = (x_i)$, on a $x_i = e_i^Tx$.

Soit $A = (a_{ij})$, on a $a_{ij} = e_i^TAe_j$.

Dans le cas de vecteurs complexes, le produit scalaire hermitien est défini par

$$(x, y) \in \mathbb{C}^n \times \mathbb{C}^n \quad \rightarrow \quad x^*y = \sum_{i=1}^{n} \bar{x}_iy_i,$$

où le surlignage d'une grandeur indique qu'on en considère le conjugué.

Il est possible de définir plusieurs normes dans l'espace vectoriel $\mathbb{R}^n$.

Définition 3.1 *Une norme d'un espace vectoriel E est une application $\|.\|$ de E dans $\mathbb{R}_+$ qui vérifie les propriétés suivantes :*

$$\begin{aligned}
&\forall x \in E,\ \|x\| \geq 0,\\
&\forall \lambda \in \mathbb{R},\ x \in E,\ \|\lambda x\| = |\lambda|\|x\|,\\
&\forall x, y \in E,\ \|x+y\| \leq \|x\| + \|y\|.
\end{aligned}$$

Dans $\mathbb{R}^n$, les trois normes les plus courantes sont la norme infinie, la norme 1 et la norme euclidienne.

- *norme infinie :* $\|x\|_\infty = \max_{i=1,\cdots,n} |x_i|$,
- *norme 1 :* $\|x\|_1 = \sum_{i=1}^n |x_i|$,
- *norme 2 ou norme euclidienne :* $\|x\|_2 = \sqrt{x^T x} = \sum_{i=1}^n {x_i}^2$

où $x = (x_1, \cdots, x_n)^T$.

La norme euclidienne est donc définie par le produit scalaire $x^T y$.

Les vecteurs de la base canonique sont normés : $\|e_j\|_2 = 1$.

Proposition 3.1 *Inégalité de Cauchy-Schwarz :*

$$\forall (x, y) \in \mathbb{R}^n \times \mathbb{R}^n,\ |x^T y| \quad \leq \quad \|x\|_2 \|y\|_2. \tag{1}$$

l'égalité a lieu si et seulement si les vecteurs x et y sont liés.

Preuve. Laissée en exercice. ◇

Comme toutes les normes d'un espace de dimension finie, ces trois normes sont équivalentes. Les constantes qui les relient sont données dans la proposition suivante.

Proposition 3.2 *Pour tout vecteur $x \in \mathbb{R}^n$, on a les inégalités :*

$$\|x\|_2 \leq \|x\|_1 \leq \sqrt{n}\|x\|_2, \tag{2}$$
$$\|x\|_\infty \leq \|x\|_2 \leq \sqrt{n}\|x\|_\infty, \tag{3}$$
$$\|x\|_\infty \leq \|x\|_1 \leq n\|x\|_\infty. \tag{4}$$

Preuve. Laissée en exercice. ◇

4 Normes matricielles

Définition 4.1 *On suppose que l'on a choisi une norme dans chacun des deux espaces $\mathbb{R}^n$ et $\mathbb{R}^m$. On définit alors la norme matricielle subordonnée dans l'espace des matrices $\mathbb{R}^{m\times n}$ par*

$$\forall A \in \mathbb{R}^{m\times n},\ \|A\| = \max_{\|x\|=1} \|Ax\|.$$

Lorsque les normes 1, 2 ou infinie sont respectivement choisies pour les deux ensembles à la fois, on note les normes subordonnées correspondantes de la même manière.

Proposition 4.1 *Soit $A = (a_{ij}) \in \mathbb{R}^{m\times n}$. Alors :*

$$\|A\|_1 = \max_{j=1,\cdots,n} \sum_{i=1}^{m} |a_{ij}|, \tag{5}$$

$$\|A\|_\infty = \max_{i=1,\cdots,n} \sum_{j=1}^{m} |a_{ij}|. \tag{6}$$

Preuve. Laissée en exercice. ◇

Remarque 4.1 *La norme matricielle euclidienne n'est pas simple à calculer dans le cas général, contrairement aux autres normes. Voir le chapitre sur les valeurs singulières.*

Proposition 4.2 *Dans certains cas particuliers, la norme des matrices est connue.*

$$\|I\|_2 = 1$$
$$\|D\|_2 = max_i|d_i|$$

Définition 4.2 *Une norme matricielle de $\mathbb{R}^{n\times n}$ est une norme qui vérifie*

$$\forall A, B \in \mathbb{R}^{n\times n},\ \|AB\| \leq \|A\|\|B\|.$$

Proposition 4.3 *Les normes subordonnées sont des normes matricielles.*

La norme de Frobenius est la norme prise au sens de l'espace vectoriel de dimension mn.

Définition 4.3 *Pour toute matrice $A = (a_{ij}) \in \mathbb{R}^{n\times m}$, on définit sa norme de Frobenius par :*

$$\|A\|_F = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{m} {a_{ij}}^2},$$
$$= \sqrt{tr(A^T A)},$$

où la trace d'une matrice est égale à la somme de ses éléments diagonaux.

Proposition 4.4 *La norme de Frobenius est une norme matricielle. Elle vérifie les inégalités suivantes*

$$\begin{aligned} \|A\|_2 &\leq \|A\|_F \leq \sqrt{n}\|A\|_2, \\ \|AB\|_F &\leq \|A\|_F\|B\|_2, \\ \|AB\|_F &\leq \|A\|_2\|B\|_F, \\ \|AB\|_F &\leq \|A\|_F\|B\|_F, \end{aligned}$$

pour tout couple de matrices $(A, B) \in \mathbb{R}^{n\times m} \times \mathbb{R}^{m\times p}$.

Preuve. Laissée en exercice. ◇

5 Orthogonalité dans $\mathbb{R}^n$

Définition 5.1 $x \perp y \Leftrightarrow x^T y = 0$

Définition 5.2 $\cos(angle(x, y)) = \frac{x^T y}{\|x\|_2\|y\|_2}$

Proposition 5.1 *Théorème de Pythagore :*
Si $x \perp y$, *alors* $\|x + y\|_2^2 = \|x\|_2^2 + \|y\|_2^2$.

Définition 5.3 *Soit* S *un sous-espace vectoriel de* $\mathbb{R}^n$. *L'orthogonal de* S *est défini par*

$$S^\perp = \{y / y^T x = 0, \quad \forall x \in S\}.$$

Proposition 5.2 $S^\perp$ *est un sous-espace vectoriel et les sous-espaces* S *et* $S^\perp$ *sont supplémentaires.*

Définition 5.4 $U = (u_1 \cdots u_k) \in \mathbb{R}^{n\times k}$, $k \leq n$ *est un système orthonormé ssi* $u_i^T u_j = \delta_{ij}$ *ssi* $U^T U = I_k$.

Remarque 5.1 *Attention, si* $k < n$, *on a* $UU^T \neq I_n$.

Preuve. Le sous-espace $U^\perp$ est non trivial, il existe donc v non nul tel que $U^T v = 0$ donc $UU^T v = 0 \neq v$. ◇

Proposition 5.3 *Un système orthonormé forme un système libre.*

Remarque 5.2 $(e_1, \ldots, e_n)$ *est une base orthonormée de* $\mathbb{R}^n$.

Proposition 5.4 *Théorème de la base incomplète. Soit* U *un système orthonormé de taille* k. *On peut compléter* U *par* U_1 *de taille* $n - k$, *pour former une base orthonormée de* $\mathbb{R}^n$. *Le système* U_1 *est une base orthonormée de* $U^\perp$. *Alors* $U_1^T U_1 = I_{n-k}$ *et* $U^T U_1 = 0$. *Tout vecteur* x *s'écrit*

$$x = UU^T x + U_1 U_1^T x.$$

Proposition 5.5 *$U \in \mathbb{R}^{n\times k}$, $k \leq n$ système orthonormé, alors*

$$\begin{array}{l}\|U\|_2 = 1.\\ \forall x \in \mathbb{R}^k, \quad \|Ux\|_2 = \|x\|_2,\\ \forall A \in \mathbb{R}^{k\times p}, \|UA\|_2 = \|A\|_2.\end{array}$$

Attention, si $k < n$, on a $\|AU\|_2 \neq \|A\|_2$.

Preuve. $\|Ux\|_2^2 = (Ux)^T(Ux) = x^T(U^TU)x = x^Tx = \|x\|_2^2$
$\|UA\|_2 = \max_{\|x\|_2=1} \|UAx\|_2 = \max_{\|x\|_2=1} \|Ax\|_2 = \|A\|_2$. ◇

Définition 5.5 *Une matrice carrée $Q \in \mathbb{R}^{n\times n}$ est orthogonale ssi $Q^TQ = I_n$. Les colonnes de Q forment une base orthonormée de $\mathbb{R}^n$.*

Proposition 5.6 *$Q \in \mathbb{R}^{n\times n}$ matrice orthogonale, alors Q est inversible, Q^T est orthogonale et*

$$\begin{array}{l}Q^TQ = QQ^T = I_n, \ Q^{-1} = Q^T.\\ \|Q\|_2 = 1.\\ \forall A \in \mathbb{R}^{n\times p}, \|QA\|_2 = \|A\|_2,\\ \forall A \in \mathbb{R}^{m\times n}, \|AQ\|_2 = \|A\|_2.\end{array}$$

Preuve. $\|AQ\|_2 = \max_{\|x\|_2=1} \|AQx\|_2 = \max_{\|y\|_2=1} \|Ay\|_2 = \|A\|_2$ car Q est inversible $(\forall y, \exists x, Qx = y)$ et $\|Qx\|_2 = \|x\|_2$. ◇

Proposition 5.7 *$Q \in \mathbb{R}^{n\times n}$ avec $\|Q\|_2 = 1$, alors Q est orthogonale. Autrement dit, une matrice carrée qui conserve les normes est orthogonale.*

Preuve. $(Qx)^T(Qx) = x^T(Q^TQ)x = x^Tx$ donc $(Q(x+y))^TQ(x+y) = (x+y)^T(x+y)$ et $(Qx)^T(Qy) = x^Ty$.

D'où $e_i^T(Q^TQ)e_i = 1$ et $e_j^T(Q^TQ)e_i = 0$, $i \neq j$ donc $Q^TQ = I_n$ et Q est orthogonale. ◇

6 Image, noyau, rang d'une matrice

$A \in \mathbb{R}^{m\times n}$.

Définition 6.1 $Im(A) = \{y \in \mathbb{R}^m / y = Ax\} = Vect(a_1, a_2, \cdots, a_n)$
$ker(A) = \{x \in \mathbb{R}^n / Ax = 0\}$

Proposition 6.1 *$Im(A)$ est un sev de $\mathbb{R}^m$ et $ker(A)$ est un sev de $\mathbb{R}^n$.*

Définition 6.2 $rang(A) = dim(Im(A))$.

Proposition 6.2 *$Im(A)^\perp = ker(A^T)$ et $Im(A^T) = ker(A)^\perp$.*

Preuve. $y \in Im(A)^\perp \Leftrightarrow \forall x, \quad (Ax)^Ty = 0 \Leftrightarrow \forall x, \quad x^T(A^Ty) = 0 \Leftrightarrow A^Ty = 0 \Leftrightarrow y \in ker(A^T)$. ◇

Proposition 6.3 $rang(A) = rang(A^T)$
$dim(ker(A)) + rang(A) = n$
$dim(ker(A^T)) + rang(A^T) = m$
$rang(A) \leq min(m, n)$.

Preuve. Soit $r = rang(A^T)$, on va montrer que $rang(A) \geq r$. Par transposition, on en déduira que $r = rang(A)$.
Soit $X = \{x_1, \ldots, x_r\}$ une base de $Im(A^T)$ et $Y = AX$. Par construction, $Y \in Im(A)$. On va montrer que Y est un système libre, ce qui implique que $r \leq rang(A)$. Pour cela, on va montrer que $Yv = 0 \Rightarrow v = 0$.

$Yv = 0 \Rightarrow AXv = 0 \Rightarrow Xv \in ker(A)$. Or $ker(A) = Im(A^T)^\perp$ donc $Xv \in Im(A^T)^\perp$. Mais X est une base de $Im(A^T)$ donc $Xv \in Im(A^T)$. Donc $Xv = 0$ et puisque X est une base, $v = 0$.

D'après la proposition précédente, on a $dim(ker(A)) + rang(A^T) = n$, on en déduit l'égalité avec $rang(A)$. ◇

Proposition 6.4 $rang(A) = n \Leftrightarrow ker(A) = \{0\}$.

Preuve. Évident d'après ce qui précède.
◇

Définition 6.3 *Soit $A \in \mathbb{R}^{m \times n}$ une matrice rectangulaire avec $m \geq n$; A est dite de rang plein si $rang(A) = n$.*

Proposition 6.5 $rang(AB) \leq rang(A),\ rang(AB) \leq rang(B)$

Preuve. $ker(B) \subset ker(AB)$ donc $rang(AB) \leq rang(B)$.
$rang((AB)^T) = rang(B^T A^T) = rang(AB) \leq rang(A^T) = rang(A)$. ◇

6.1 Matrices de rang k

Proposition 6.6 *Soit $U = (u_1 \ldots u_k)$ et $V = (v_1 \ldots v_k)$ deux systèmes libres de $\mathbb{R}^m$ et de $\mathbb{R}^n$ respectivement, alors $UV^T \in \mathbb{R}^{m \times n}$ et*

$$ker(UV^T) = V^\perp,\ Im(UV^T) = Vect(U),\ rang(UV^T) = k.$$

Réciproquement, si A est une matrice de rang k, il existe $U = (u_1 \ldots u_k)$ base de $Im(A)$ et $V = (v_1 \ldots v_k)$ base de $ker(A)^\perp$ tels que $A = UV^T$.

Preuve. Soit $A = UV^T$, alors $x \in ker(A) \Leftrightarrow UV^T x = 0 \Leftrightarrow V^T x = 0 \Leftrightarrow x \in V^\perp$ donc $ker(A) = V^\perp$. On en déduit que $rang(A) = k$ car $dim(V^\perp) = n - k$. D'autre part, $Im(A) \subset Im(U)$ d'où, par égalité des dimensions, $Im(A) = Im(U)$.

Réciproquement, soit $V \in \mathbb{R}^{n \times k}$ une base orthonormée de $ker(A)^\perp$, complétée par V_1 dans $ker(A)$. Alors $\forall x,\ x = VV^T x + V_1 V_1^T x$ et $Ax = AVV^T x$. Soit $U = AV$, alors $Ax = UV^T x$ donc $A = UV^T$. De plus, U est un système libre car $Uy = 0 \Leftrightarrow AVy = 0 \Leftrightarrow Vy \in ker(A) \cap ker(A)^\perp \Leftrightarrow y = 0$, donc U est une base de $Im(A)$. ◇

6.2 Matrices de rang 1

En particulier, les matrices de rang 1 sont de la forme xy^T, où x et y sont deux vecteurs non nuls.

Proposition 6.7 *Si $v^T u \neq 0$, la transformation $P = \frac{1}{v^T u} uv^T$ est la projection sur la droite engendrée par le vecteur u orthogonalement au vecteur v. Si de plus $u = v$, la projection P est une projection orthogonale.*

Preuve. Il est évident que $Pu = u$ et $Pw = 0$ pour tout $w \perp v$. Puisque $\mathbb{R}^n = (u) \oplus (v)^\perp$, la projection P est caractérisée. Si $u = v$, la projection $P = \frac{1}{\|u\|_2{}^2} uu^T$ est la projection orthogonale sur (u). ◇

Remarque 6.1 *Si $u \perp v$, alors la transformation $N = \alpha uv^T$ est nilpotente, de noyau l'hyperplan orthogonal à v.*

Proposition 6.8 *Si $v^T u \neq 0$, la transformation $Q = I - \frac{1}{v^T u} uv^T$ est la projection sur l'hyperplan $(v)^\perp$ parallèlement à la droite engendrée par le vecteur u. Si de plus $u = v$, la projection Q est la projection orthogonale sur l'hyperplan $(u)^\perp$.*

Preuve. Évident. ◇

7 Notions de complexité et de performances

Les algorithmes de calcul sont caractérisés par leur complexité arithmétique, mesurée par le nombre d'opérations (additions, multiplications, etc) sur des réels, ainsi que par leur coût de stockage, mesuré par le nombre de variables réelles. Le stockage des nombres entiers et les calculs sur les nombres entiers sont d'un coût marginal, qui est ignoré. En pratique, les opérations arithmétiques et le stockage se font sur des nombres flottants (voir chapitre sur l'arithmétique flottante). Les opérations inutiles, telles qu'une multiplication par zéro ou par un, ne sont pas comptabilisées.

Les performances d'un algorithme se mesurent par sa vitesse de calcul. Celle-ci dépend de la complexité mais aussi de l'exploitation de l'architecture de l'ordinateur, notamment du parallélisme interne et de la hiérarchie des mémoires.

Pour la complexité, on donnera souvent le terme prédominant, du plus grand ordre. Pour une complexité polynomiale, ce terme est écrit sous la forme $O(n^k)$. Cela signifie que le nombre d'opérations N divisé par n^k tend vers une constante quand n tend vers l'infini.

La plupart des algorithmes d'algèbre linéaire ont une complexité polynomiale, par exemple $N = an^3 + bn^2 + cn + d$. On a pour cet exemple $N = O(n^3) = an^3 + O(n^2)$.

8 Bibliothèques BLAS et LAPACK

Les opérations de base d'algèbre linéaire sont regroupées dans une bibliothèque numérique appelée BLAS : Basic Linear Algebra Subroutines. Cette bibliothèque est souvent fournie par le constructeur et optimisée pour une architecture donnée. Les opérations sont divisées en trois niveaux, appelés BLAS1, BLAS2, BLAS3. L'optimisation concerne l'ordre des opérations et l'accès à la mémoire, pour exploiter au mieux la hiérarchie (mémoire principale, mémoire cache, etc). Les performances (vitesse de calcul) sont d'autant meilleures que le niveau est élevé.

Les opérations plus complexes sont regroupées dans la bibliothèque numérique appelée LAPACK : Linear Algebra Package. Les opérations de LAPACK utilisent au maximum les opérations BLAS, surtout BLAS3, qui est le plus performant en temps de calcul.

8.1 Opérations BLAS1

Ce niveau concerne les opérations entre vecteurs. Quelques exemples :

- combinaison linéaire de vecteurs $z = a * x + b * y$, $3n$ opérations.
- produit scalaire de vecteurs $a = a + x^T y$, $2n$ opérations.
- produit de matrices diagonales, n opérations.

Toutes les opérations BLAS1 ont une complexité en $O(n)$ opérations flottantes et un accès mémoire en $O(n)$ mots flottants.

Il n'y a qu'un niveau de boucle.

8.2 Opérations BLAS2

Ce niveau concerne les opérations entre matrices et vecteurs. Quelques exemples :

- produit matrice-vecteur $y = y + A * x$, $2mn$ opérations.
- produit extérieur de vecteurs $A = xy^T$, mn opérations.

Dans le cas de matrices carrées d'ordre n, toutes les opérations BLAS2 ont une complexité en $O(n^2)$ et un accès mémoire en $O(n^2)$.

Il y a deux niveaux de boucle imbriqués, ce qui permet de construire deux variantes suivant l'ordre des boucles. On peut ainsi choisir un calcul par lignes ou par colonnes. On peut aussi définir une partition de la matrice et effectuer les opérations par blocs, pour optimiser l'accès hiérarchique à la mémoire.

Soit $Q = I - \frac{1}{v^T u} uv^T$ la matrice de projection de rang 1 définie précédemment. Il est à noter que

$$Qx = x - \frac{1}{v^T u} uv^T x = x - \frac{1}{v^T u}(v^T x)u.$$

Il est inutile et coûteux de calculer la matrice Q. En effet, le produit Qx est une opération matrice-vecteur de type BLAS2 et de complexité $O(nm)$, alors que l'expression ci-dessus n'utilise que des opérations vectorielles de type BLAS1 et de complexité $O(n)$ ou $O(m)$.

8.3 Opérations BLAS3

Ce niveau concerne les opérations entre matrices. Quelques exemples :

- produit de matrices $C = C + A * B$, $2mnp$ opérations.
- produit $C = C + AA^T$, $2mn^2$ opérations.

Dans le cas de matrices carrées d'ordre n, toutes les opérations BLAS3 ont une complexité en $O(n^3)$, avec un accès mémoire en $O(n^2)$.

Les trois niveaux de boucle imbriqués permettent de définir six variantes suivant l'ordre des boucles. On peut choisir de parcourir chacune des matrices par lignes ou par colonnes. Comme dans le niveau BLAS2, on optimise l'accès à la mémoire en définissant une partition par blocs. De plus, comme l'algorithme fait plus d'opérations arithmétiques que d'accès aux données, on peut ré-utiliser des valeurs qui sont dans la mémoire cache. C'est pour cette raison que le niveau 3 est le plus performant et permet presque d'atteindre la performance maximale d'une machine.

8.4 Produit de matrices

Le produit de matrices est l'opération la plus utilisée dans BLAS3. L'analyse suivante montre comment organiser les calculs pour exploiter la hiérarchie de mémoires. On suppose que les calculs sont effectués sur un processeur qui possède une mémoire cache. Les matrices sont partitionnées par blocs et l'objectif est de déterminer la taille des blocs pour utiliser au mieux la mémoire cache. Cette étude est une adaptation au cas monoprocesseur de [2].

Soit M la taille du cache. Nous supposons que les matrices A, B et C sont respectivement de taille $n_1 \times n_2$, $n_2 \times n_3$ et $n_1 \times n_3$, et qu'elles ont été partitionnées en blocs de tailles respectives $m_1 \times m_2$, $m_2 \times m_3$ et $m_1 \times m_3$. On suppose $n_i = m_i * k_i$ pour tout $i = 1,\ 2,\ 3$. L'objet de l'étude est alors de trouver les valeurs m_i qui utilisent au mieux le cache, c'est-à-dire qui permettent le plus de réutilisation des données.

L'opération $C = A * B$ peut s'écrire par blocs

```
do i = 1, k1
  do k = 1, k2
    do j = 1, k3
      Cij := Cij + Aik * Bkj
    enddo
  enddo
enddo
```

Dans la boucle interne j, le bloc A_{ik} reste le même; on suppose donc qu'il réside dans le cache; sa taille est $m_1 m_2$. Cela entraîne la première contrainte :

$$m_1 m_2 \leq M$$

Il est évident que les blocs sont plus petits que les matrices, ce que l'on traduit par $1 \leq m_i \leq n_i$ pour $i = 1, 2, 3$. On a ainsi obtenu l'ensemble des contraintes sous lesquelles on doit minimiser les mouvements de données entre la mémoire et le cache.

Si on calcule le nombre de lectures nécessaires pour avoir en cache les variables nécessaires au produit, on trouve que la matrice A est lue une fois, la matrice B l'est k_1 fois et la matrice C l'est k_2 fois. Au total, le nombre des lectures est:

$$L = n_1 n_2 + n_1 n_2 n_3 (\frac{1}{m_1} + \frac{1}{m_2})$$

Il reste donc à trouver les valeurs m_1 et m_2 qui minimisent $\frac{1}{m_1} + \frac{1}{m_2}$ sous les contraintes précédentes. On en arrive à la politique suivante (le choix de m_3 est sans importance):

1. si $n_2 n_1 \leq M$ alors $m_1 = n_1$ et $m_2 = n_2$;
2. sinon si $n_2 \leq \sqrt{M}$ alors $m_1 = \frac{M}{n_2}$ et $m_2 = n_2$;
3. sinon si $n_1 \leq \sqrt{M}$ alors $m_1 = n_1$ et $m_2 = \frac{M}{n_1}$
4. sinon $m_1 = \sqrt{M}$ et $m_2 = \sqrt{M}$.

Preuve. Preuve du dernier cas.
$m_2 = M/m_1$ et $d/dm(1/m_1 + 1/m_2) = 0$ d'où $-1/m^2 + 1/M = 0$, soit $m = \sqrt{M}$.
◇

8.5 bibliothèque LAPACK

La bibliothèqe LAPACK regroupe la plupart des algorithmes d'algèbre linéaire. Elle contient toutes les fonctions pour résoudre les systèmes linéaires, les problèmes aux moindres carrés, la décomposition aux valeurs singulières, les problèmes de valeurs propres.

LAPACK est la meilleure référence pour l'algèbre linéaire sur matrices stockées sous format plein ou sous format bande. Elle est disponible sur le site NETLIB (http://www.netlib.org) et le manuel d'utilisation est édité [1].

LAPACK utilise une partition par blocs des matrices, de façon à exploiter les performances des opérations BLAS3. D'autre part, les algorithmes sont robustes vis-à-vis des erreurs d'arrondi et il est possible d'estimer la sensibilité aux variations des données (voir chapitre sur l'arithmétique flottante).

Les chapitres suivants décriront divers exemples d'algorithmes de la bibliothèque LAPACK.

References

[1] E. Anderson, z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK Users' Guide, Third Edition*. SIAM, 1999.

[2] K. Gallivan, W. Jalby, and U. Meier. The use of blas3 in linear algebra on a parallel processor with hierachical memory. *SIAM J. Sci. Stat. Comput.*, 1986.